filed is sufficiently clear to one skilled in the art. Therefore, Applicant respectfully requests reconsideration of the specification objection.

Claim Rejections - 35 U.S.C. § 102(e)

Claims 1, 9, and 19 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Dunn et al. (U.S. Patent No. 6,247,172) (hereinafter "Dunn"). Reconsideration of claims 1, 9, and 19 is respectfully requested.

As explained in the Background section of the instant application, conventional emulation typically uses one of two known approaches: translation or interpretation. Translation converts a program in code native to the target machine to a program in code native to the host machine (application page 2, lines 13-16). Translation may be static or dynamic. Static translation translates all of the program code of the target machine prior to any execution of the translated code file (application page 5, lines 21-22). Dynamic translation generates code native to the host machine "on the fly." Translation generally is complex, especially when the instruction sets of the target and host machines are significantly different (application page 2, lines 19-20). An interpreter, on the other hand, "interprets" each instruction of a target program and then performs equivalent operations on the host machine, without requiring translation of the code. Interpretive execution is typically much slower than the execution of native code generated through translation (application page 3, lines 1-2).

The present invention takes a novel approach to emulation that overcomes the shortcomings of conventional emulation. According to the present invention, the target program to be emulated (*e.g.*, a program comprising instructions of the Unisys "E-mode" instruction set) is first *statically translated* to a series of instructions of an *intermediate instruction set*. The intermediate instruction set is an instruction set that is *optimized for interpretation* on the host computer (*e.g.*, a computer that employs the Alpha microprocessor). The series of translated instructions is then *executed by interpretation* on the host computer (*e.g.*, an Alpha server). The intermediate instruction set is not the native instruction set of either the target program (*e.g.*, "E-mode" instruction set) or the host computer (*e.g.*, Alpha microprocessor instruction set). Because the intermediate instruction set is optimized for interpretation on the host computer, the execution of the statically translated intermediate code by interpretation on the host computer will generally be faster than conventional

interpretation.

The foregoing features of the present invention are recited in each of the independent claims. For example, claim 1 recites:

1. A method for emulating the execution of a target program comprising instructions of an instruction set of a target on a host computer *having a different instruction set*, said method comprising:

performing a *static translation* of the instructions of the target program into a series of instructions of an *intermediate instruction set*, the intermediate instruction set being *optimized for interpretation* on the host computer; and

executing the series of instructions of the intermediate instruction set by interpretation on the host computer. (emphasis added)

Independent claims 9 and 19 recite similar features.

On the contrary, Dunn is an example of a conventional emulator that performs emulation by *translating* code from a legacy system to a target system. The goal of the emulator of Dunn is to provide fault recovery while still allowing optimization in translation. The translation may be static or dynamic. Either way, a translator 34 translates from a user application 32 to optimized target code 40 and recovery blocks 42. The optimized *translated target code* 40 is then *executed* (Dunn column 5, lines 52-53) while recovery blocks 42 are used for fault recovery. Dunn does not teach or suggest first *statically translating* a target program into a series of instructions of an intermediate instructions set, wherein the *intermediate instructions set is optimized for interpretation* on a host computer, and then executing the intermediate instructions by interpretation on the host computer, as claimed by Applicants.

Accordingly, Applicant respectfully submits that claims 1, 9, and 19 are not anticipated by Dunn. Reconsideration of the Section 102(e) rejection of these claims is respectfully requested.

Claim Rejections - 35 U.S.C. § 103(a)

Claims 2-8, 10-18, and 20-31 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dunn in view of James R. Gillig, "Endian-Neutral Software, Part 2", published November 1, 1994 (hereinafter "Gillig"). Reconsideration of claims 2-8, 10-18, and 20-31 is respectfully requested.

As described in detail above, Dunn is directed to a conventional emulator that performs emulation through either static or dynamic translation of code. Gillig is directed to designing and

writing software to be "endian" neutral. Processors may use only big-endian addressing format, only little-endian addressing format, or both addressing formats. Big-endian format maps the lowest address to the highest order data byte and little-endian format maps the lowest address to the lowest order data byte. Gillig discloses recommendations for writing endian neutral software, such as using a high level language, using data types correctly, and the like. According to Gillig, if these techniques are practiced, software can more easily be ported among processors that use different endian formats.

Gillig does not, however, cure the deficiencies of Dunn. Gillig does not teach or suggest any form of emulation in which a target program to be emulated is first statically translated into a series of instructions of an intermediate instructions set, wherein the intermediate instruction set is optimized for interpretation on a host computer, and then the intermediate instructions are executed by interpretation on the host computer, as claimed by Applicants. Gillig merely describes how to write software that is easily ported among processors that use different endian formats. Thus, combining the teachings of Gillig with the teachings of Dunn would not result in the present invention as claimed.

For the foregoing reasons, Applicants respectfully request reconsideration of the Section 103(a) rejection of claims 2-8, 10-18, 20-26, and 28-31.

CONCLUSION

For the foregoing reasons, Applicants respectfully submit that all of the claims of the instant application patentably define over the prior art of record, alone or in combination. Reconsideration of the Office Action and an early Notice of Allowance are respectfully requested. In the event that the Examiner cannot allow the present application for any reason, the Examiner is encouraged to contact the undersigned attorney, Raymond N. Scott Jr. at (215) 564-8951, to discuss resolution of any remaining issues.

Respectfully submitted,

Date: 11/9/01

Raymond N. Scott Jr.
Attorney for Applicant
Registration No. 48,666

WOODCOCK WASHBURN LLP
One Liberty Place - 46th Floor
Philadelphia, Pennsylvania 19103
Telephone: 215.568.3100
Facsimile: 215.568.3439